THE MULTIPLE REPRESENTATION PROBLEM IN GENETIC APPROACHES FOR INDEX ASSIGNMENT IN VECTOR QUANTIZATION CODEBOOK DESIGN

Armando Malanda-Trigueros(*), Jesús-M. Val-Bueno(*), Aníbal-R. Figueiras-Vidal (**)

(*) Universidad Pública de Navarra D.I.E.E. Campus de Arrosadía s/n. 31.006 Pamplona. Spain.
Tel:+34 48 169312. Fax: +34 48 169720 e-mail: malanda@unavarra.es
(**) Universidad Carlos III de Madrid. A.T.S.C. DI-EPS. C/ Butarque, 15. 28.911 Leganés. Madrid. Spain.
Tel:+34 1 6249923. Fax:+34 1 6249430. e-mail: arfv@ing.uc3m.es

Abstract:
In Vector Quantization codebook design for noisy channels, binary indexes have to be assigned to the different codevectors. A multiple representation problem arises in this assignment. This problem is explained and some solutions presented. Implementation of these solutions in a Genetic-based approach is also commented.

Index terms:
Vector Quantization, index assignment, multiple representation problem, Genetic Algorithms.

I. INTRODUCTION

In the last two decades data compression by means of Vector Quantization (VQ) has received enormous attention, motivated by the expansion of telecommunication technologies and services [1]. The design for VQ codebooks that are efficient for data compression and robust against channel errors has become fundamental [2]. To face this dual problem (codebook design + index assignment) some methods deal with either constituent separately [3], [4], [5]. Others use integrated approaches in which the codebook search conveys the index assignment only in an implicit manner [6]. When index assignment optimisation is addressed, a multiplicity of solutions appear. In some techniques this multiplicity means a considerable drawback. This is particularly the case with Genetic Algorithms (GA) [7]. In this paper this multiple representation problem is explained and simple solutions provided. In [8], a GA for VQ design was presented. We have modified that algorithm to include the mentioned solutions for the multiple representation problem. This new approach is explained in this paper, and a simulation study in which three grey images are vector quantized is reported. The results of the study are commented and finally some conclusions extracted.

II. FUNDAMENTALS

A. Vector Quantization

In VQ (Figure 1) a source vector $\bar{x}$ of dimension k is coded into index i, meaning that vector $\bar{c}_i$ has been selected as the best representative of vector $\bar{x}$ within the L-sized codebook $\{C\}=\{\bar{c}_0,\bar{c}_2,..,\bar{c}_{L-1}\}$. This selection is usually called coding rule. The binary representation of index i is $b_\pi^{(i)}$ obtained as the output of the index

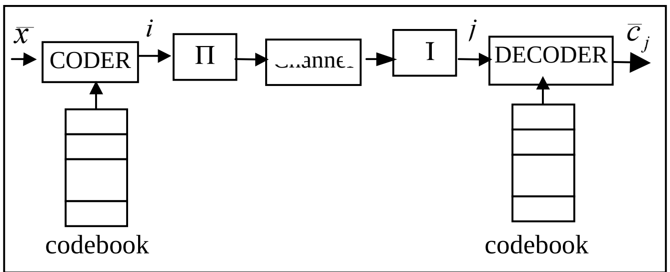

Figure 1. VQ-based system

assignment function (i.a.f.) π to input i. Then $b_\pi^{(i)}$ traverses a noisy channel at the end of which, $b_\pi^{(j)}$ is received. This binary $b_\pi^{(j)}$ is decoded into index j, which in turn, points to vector $\bar{c}_j$ as its final output. The problem of VQ design is that of finding the best codebook, coding rule and assignment function, so that $\bar{c}_j$ is statistically as close to $\bar{x}$ as possible, i.e., the system distortion is minimised. If the training vector set

$V = \{\bar{v}_0, \bar{v}_1, \ \bar{v}_{N-1}\}$ embodies the statistical distribution of $\bar{x}$, this distortion may be approximated by [6].

$$D(\{C\},\{S\},\pi) = \frac{1}{N}\sum_{i=0}^{L-1}\sum_{k=0}^{N_i-1}\sum_{j=0}^{L-1}\Pr\left(b_\pi^{(i)} / b_\pi^{(j)}\right)\left|\bar{v}_{ik} - \bar{c}_j\right|_2^2 \qquad (1)$$

where $\{S\} = \{S_1, S_2, .., S_{L-1}\}$ is the partition of V formed by the regions $S_i = \{\bar{v}_{i0}, \bar{v}_{i1}, \ , \bar{v}_{iN_i-1}\}$ (subsets of V comprised by the training vectors that are coded into index i), N_i is the cardinal of S_i, and $\Pr\left(b_\pi^{(j)} / b_\pi^{(j)}\right)$, the probability of receiving $b_\pi^{(j)}$ when $b_\pi^{(i)}$ was actually transmitted ($\left|\bar{a} - \bar{b}\right|_2^2$ calls for the square Euclidean distance between any two vectors $\bar{a}$ and $\bar{b}$).
The channel is considered to be additive, memoryless and binary symmetric with a bit error rate (BER) of value ε. Index transition probabilities are given by [4]:

$$\Pr\left(b_\pi^{(i)} / b_\pi^{(j)}\right) = \varepsilon^m (1-\varepsilon)^{\psi_{ij} - m} \qquad (2.1)$$

$$\psi_{ij} = ham\left(b_\pi^{(i)}, b_\pi^{(j)}\right) \qquad (2.2)$$

where m is the number of bits to code i and j, ($m = \log_2(L)$) and $ham(b_1, b_2)$, the Hamming distance (number of distinct bits) between any two binaries b_1 and b_2.

Given a codebook, an assignment function and a probabilities scheme, the best coding strategy is given by the Generalised Nearest Neighbour Rule (GNN) [6]. Besides provided an assignment function, a probabilities scheme, and a coding strategy given by a certain partition, the best codebook is stated by the Generalised Centroid Rule (GC) [6]. The alternate and iterative application of the GNN and the GC rules comprises the GLA algorithm [4], which produces a monotonic decrease of distortion usually converging to a local minimum system distortion point in the codebook space. There is not a known third rule to optimise the index assignment function (i.a.f.) Π explicitly; here a GA will be used for this task.

B. Genetic Algorithms

GA are global optimisation procedures inspired in Nature that keep a population of tentative solutions (called individuals or chromosomes) to a given problem. These individuals evolve in such a way that better solutions are achieved as the algorithm progresses (see Figure. 2). This evolution is based on selection of the fittest individuals in the population, together with several random procedures such as crossover (offspring generation from two selected individuals) and mutation (random change in one individual) [9].

Step 1. Initialization.

Step 2. Cost function evaluation.

Step 3. IF convergence has been reached or a given number of generations have taken place, HALT.

Step 4. Selection of the fittest individuals.

Step 5. Procreation of new individuals from the selected ones.

Step 6. GO TO Step 2

Figure 2. Genetic Algorithm

III. INDEX ASSIGNMENT AND THE MULTIPLE REPRESENTATION PROBLEM

A. The problem

The i.a.f. π can be expressed by means of the matrix M_π:

$$M_\pi = \begin{bmatrix} b_\pi^{(0)} \\ b_\pi^{(1)} \\ \\ b_\pi^{(L-1)} \end{bmatrix} \qquad (3)$$

The i-th row of this matrix, namely $b_\pi^{( )}$, contains a string with the m bits with which codevector $\bar{c}_i$ is coded. If $\{C\}$ and $\{S\}$ are given, the distortion in (1) is only dependent on π, which only affects the term $\Pr\left(b_\pi^i / b_\pi^j\right)$. Moreover, as it can be inferred from (2), π only affects this probability of error by way of $ham\left(b_\pi^i, b_\pi^j\right)$. From this comes that if two different i.a.f. (π_1 and π_2) are such that

$$ham\left(b_{\pi_1}^i, b_{\pi_1}^j\right) = ham\left(b_{\pi_2}^i, b_{\pi_2}^j\right)$$
$$\forall i, j = 0, 1, \ .., L-1. \qquad (4)$$

they will render identical mean distortion. Therefore they can be considered as “equivalent” (i.e., belonging to the same class of i.a.f.). The following two factors are the cause of the multiplicity in the i.a.f. space [7]:

f$_1$- If M_{π_1} is equal to M_{π_2} with the sole difference of one or several columns in which the zeros in M_{π_1} are ones in M_{π_2} and vice-versa, then these two matrixes will hold condition (4) and, π_1 and π_2 will be equivalent i.a.f. This stems from the symmetry of the channel BER with respect to zeros and ones. As every column can be taken in two different ways there exist $2^m = L$ different and equivalent i.a.f. for a given π.

f$_2$- If M_{π_2} is obtained from matrix M_{π_1} permuting two or more of its columns, it results that M_{π_1} and M_{π_2} will attend condition (4). Again π_1 and π_2 will be equivalent i.a.f. The demonstration of this property comes from the equal probability of error in the m different bits composing the bit string of a codevector. A total of m! different equivalent i.a.f. can be found in such a way.

As the two previous transformations do not interfere (if from matrix B_{π_1}, the new matrix B_{π_2} can be reached by means of f$_1$, it cannot be reached by means of f$_2$ and vice-versa), it results that any i.a.f. class contains at least $2^m m!$ members [7]. For many optimisation methods this multiplicity of solutions represents an important inconvenience. This is particularly true with GA, as different individuals will move towards different minima, and the population will not co-operate as a whole in the direction of only one minimum point. It would thus be desirable that each class could have a representative member.

B. The solutions

s$_1$- The first multiplicity factor (f$_1$) can be dealt with by imposing that the last row in every matrix M_π be formed only by ones:

$$b_\pi^{(L-1)} = \{1, 1, .., 1\} \tag{5}$$

This way, no one-to-zero or zero-to-one column interchange is permitted. This is something already considered in [7].

s$_2$- The second multiplicity factor (f$_2$) can be deterred in the following way. Let establish a measure $Q_\pi(s)$ for the column s of matrix M_π, for example, transforming this binary column vector into an integer representation given by:

$$Q_\pi(s) = \sum_{t=0}^{L-1} B_\pi(t,s)\, 2^t \tag{6}$$

where $B_\pi(t,s)$ is the s-th entry to binary vector $b_\pi^{(t)}$. Let then proceed to permute all the columns of M_π, reordering them according to their $Q_\pi(s)$ measurement. Call this new matrix $\hat{M}_\pi$. Any matrix belonging to the same class as M_π and suffering these operations will turn into the same matrix $\hat{M}_\pi$. Therefore $\hat{M}_\pi$ can be considered as the representative matrix of this class.

In [7] solution s$_1$ was suggested to deal with factor f$_1$, but f$_2$ was left unattended. Applying s$_1$ and s$_2$, the multiplicity representation will collapse and the search space will be reduced in a factor of $2\ \ m!$.

IV. THE HYBRID GENETIC ALGORITHM

Based in a previous work [8] a new VQ design method is proposed in which GLA is used to find good codebooks and a specially designed GA (which we call GAIA) tries to obtain good index assignments for these codebooks. The new Hybrid Genetic Algorithm (HGA) maintains a collection of individuals, each one consisting of:
-a tentative codebook,
-a tentative index assignment for this codebook.
The overall HGA consists on the alternate and iterative application of GLA and GAIA, as expressed in Figure 3. The GLA operates only on the codevectors, leaving the index assignments unaffected. First a number (T) of iterations are executed to each individual, and their codebooks changed accordingly. Then S iterations of GAIA are run and the individuals' index assignments evolve, without altering the previously formed codebooks. This process is repeated as far as the system distortion keeps reducing.

Step 1. Codebook initialisation.

Step 2 T iterations of GLA.

Step 3. S iterations of GAIA.

Step 4. IF mean distortion is reduced, GO TO Step 2.

Step 5. OTHERWISE,

HALT.

Figure.3. Hybrid Genetic Algorithm

A. Genetic Algorithm for Index Assignment

In [8] a GA was used to find index assignments in a VQ optimisation process. Here we have modified that algorithm and include the two solutions for the multiple representation problem explained in Section 3. Our GAIA is a GA, in which individuals are tentative i.a.f. (π_j , j=1,2, ,M). Let $e_{\pi_j}^{(i)}$ be the integer representation of $b_{\pi_j}^{(i)}$. This way π_j is represented by the list $B_{\pi_j} = \{e_{\pi_j}^{(0)}, e_{\pi_j}^{(1)}, .., e_{\pi_j}^{(L-1)}\}$, which will be one of the $L!$ possible permutations of the list $\{0,1, \, .. \, L\text{-}1\}$. In GAIA, the following genetic operators are used:

- Evaluation: each individual is evaluated and its fitness obtained from the distortion function in (1) as:

$$fitness(\pi_j) = \underbrace{max}_{k=1,..,M} \{D(\{C\},\{S\},\pi_k)\} - D(\{C\},\{S\},\pi_j). \qquad (7)$$

- Random selection proportional to individual's fitness [9];
- Cycle crossover: as described in [10];
- Mutation: with a certain probability, for every individual vector, two randomly selected entries are interchanged.
- Reordering 1: s_1 is included (i.e., throughout all the space of solutions, $e_{\pi_j}^{(L-1)} = L-1$).
- Reordering 2: application of solution s_2 after every crossover and mutation operation.
- Repetition removal: whenever two individuals of the population are identical, one of them is mutated.
- Elitism: the best solution in one generation is preserved for the next generation without being crossed or mutated [9].

V. SIMULATION TESTS AND RESULTS

A. Simulation tests

We used for our tests Lena, Baboon and Pepper images (256x256 pixels large, initially quantized at 8 bits/pixel). 4096 non-overlapping 16-dimensional vectors, corresponding to 4x4 pixel blocks, were taken as the training sets for each image. Finally, we used codebooks of sizes 64 and 128, and BERs of 1.0E-4, 1.0E-3 and 1.0E-2.

The following algorithms were tested for comparison:

a- HGA I: Our HGA as explained before.
b- HGA II: HGA in which the crossover operator was removed
c- HGA III: HGA in which the mutation operator was removed
d- HGA IV: HGA in which the reordering 2 operator (s2) was removed.
e- GLA: as described in [4] and outlined in Section II.A.

As merit figure we used the widespread Peak Signal to Noise Ratio, defined as [3]:

$$PSNR = 10 \log_{10}(255^2 \, k/D) \qquad (8)$$

being D the distortion in (1) and k, the vector dimension. For each one of these test cases several independent runs were carried out until a total amount of 20.000 GLA simple iterations (GNN + GC rules) were processed. The minimum distortion throughout all these runs was taken as the term D in (8). This way the different algorithms were tested in a computationally fair basis.

B. Results

In Table 1 and 2 results of these tests are reported.

	Lena			**Baboon**			**Pepper**		
BER	**1.0E-4**	**1.0E-3**	**1.0E-2**	**1.0E-4**	**1.0E-3**	**1.0E-2**	**1.0E-4**	**1.0E-3**	**1.0E-2**
HGA I	26.65	26.58	25.17	23.67	23.57	22.98	27.42	27.01	25.41
HGA II	26.65	26.56	25.10	23.44	23.51	22.96	27.35	27.05	25.44
HGA III	26.63	26.47	25.08	23.47	23.49	22.96	27.38	27.05	25.44
HGA VI	26.60	26.53	25.16	23.47	23.50	22.96	27.33	27.02	25.39
GLA	26.95	26.30	24.89	23.66	23.49	22.84	27.42	26.81	25.17

Table 1. *PSNR* of the VQ of Lena, Baboon and Pepper images with codebooks of size *L*=64.

	Lena			**Baboon**			**Pepper**		
BER	**1.0E-4**	**1.0E-3**	**1.0E-2**	**1.0E-4**	**1.0E-3**	**1.0E-2**	**1.0E-4**	**1.0E-3**	**1.0E-2**
HGA I	27.57	27.56	25.99	24.3	24.20	23.51	28.71	28.21	26.42
HGA II	27.52	27.51	25.94	23.82	24.05	23.42	28.35	28.11	26.38
HGA III	27.54	27.45	25.94	23.83	24.00	23.42	28.29	28.07	26.23
HGA VI	27.53	27.49	25.99	23.89	24.05	23.43	28.31	28.13	26.30
GLA	27.95	27.17	25.60	24.31	24.09	23.27	28.61	27.76	25.98

Table 2. *PSNR* of the VQ of Lena, Baboon and Pepper images with codebooks of size *L*=128.

They show a general superiority of our HGA over GLA, (only for Lena image and a BER of 1.0E-4 this does not hold). It is also important to point out the clear tendency of the Hybrid approach to improve performance as the BER grows and also as the codebook size increases, as compared to GLA. Finally the slight but consistent decay in system performance when crossover, mutation or the reordering operator are removed proves the convenience of the three of them. Particularly the benefit in the use of the reordering operators calls for the attention that the multiple representation problem should be given and serves to indicate the successful use of mechanisms s_1 and s_2 to deal with it.

V. CONCLUSIONS

The multiple representation problem, which appears in index assignment optimisation in the context of noisy channel VQ design has been explored. Simple solutions have been provided. These solutions have been included in the genetic part of a hybrid (Genetic + GLA) VQ design method. Several tests in the VQ of grey images confirm the convenience of separated and alternated strategies for the codebook + index assignment search, and also reinforce the confidence in approaches that deal with the multiple representation problem as the ones suggested here, particularly for the VQ design for highly noisy channels.

References

[1] V. Bhaskaran and K. Konstantinides. "Image and Video Compression Standards. Algorithms and Architectures (2nd. Ed.)". Kluwer Academic Publishers. Boston, 1997.

[2] A. Gersho and M. R. Gray. "Vector Quantization and Signal Compression". Kluwer Academic Publishers. Boston, 1992.

[3] K. A. Zeger and A. Gersho. “Zero redundancy channel coding in vector quantization” . IEEE Electronic Letters, vol.23, no.12, Jun.1987. pp.654-655.

[4] N. Farvardin. “A study of Vector Quantization for noisy channels”. IEEE Trans. on Info. Theory, vol. 36, no.4 July 1990. pp. 799-809.

[5] V. Cuperman, F. H. Liu and P. Ho. “Robust Vector Quantization for noisy channels using Soft Decision and Sequential Decoding”. European Trans. on Telecommunications and Related Technologies. Vol 5 no. 5 Sept.-Oct. 1994. pp. 541-552.

[6] D. Miller and K. Rose. “Combined Source-Channel Vector Quantization using Deterministic Annealing”. IEEE Trans. Comm., vol. 42, no.2/3/4 Feb/March/April. 1994. pp. 347-356..

[7] J.S. Pan, F. R. McInnes and M.A. Jack. “Application of parallel Genetic Algorithm and property of multiple global optima to VQ codevector index assignment for noisy channels”. Electronics Letters on Line. 15 February 1996 (No. 19960194).

[8] A. Malanda and A.R. Figueiras-Vidal, "Hybrid Genetic Algorithm for Codebook Enhancement in Noisy Channel Vector Quantization". Procc. of the IASTED Signal Processing and Communications Conf. pp. 511-513. Las Palmas de Gran Canaria, Feb. 1998.

[9] D. E. Goldberg. “Genetic Algorithms in Search, Optimization and Machine Learning”. Addison-Wesley, Reading MA. 1989.

[10] I. M. Oliver, D. J. Smith and J. Holland, “A Study of Permutation Crossover Operations on the Travelling Salesman Problem”. In Procc. of Int. Conf. On GA. Hillsdale, NJ, 1987. pp. 224-230.